DESIGNING DIELECTRIC GRATING FILTERS WITH PGAPACK

Cinzia Zuffada, David Levine* and Sorin Tibuleac**

Jet Propulsion Laboratory

California Institute of Technology, Pasadena, CA 91109

* *The Boeing Company*

Seattle, WA 98124

** *The University of Texas at Arlington*

Department of Electrical Engineering, Arlington, TX 76019

OUTLINE

CHAPTER SYNOPSIS

The chapter is concerned with the design of planar dielectric layer diffraction gratings, which exhibit sharp resonances due to the coupling of exterior evanescent diffractive fields to the leaky modes of dielectric waveguides. In these cases efficient switching of energy between (nearly) totally reflected zero-order mode and (nearly) totally transmitted zero-order mode is achieved over a small variation in wavelength, with proper choice of the cell size. Such property leads to the possibility of filter designs whose arbitrarily narrow linewidths can be controlled by the choice of modulation amplitude and mode confinement. Adopting an inverse approach, the feasibility of novel designs in this class of devices is demonstrated, using the genetic algorithm library PGAPACK to solve for material dielectric constants and geometric boundaries defining homogeneous regions of the periodic cell. In particular, genetic algorithms show that simple geometries, not previously reported, utilizing a small number of layers and/or gratings, can be found to yield bandpass or stopband filters with user defined linewidth at microwave frequencies. The evaluation of the fitness of a candidate design entails the solution of the forward scattering problem for the grating. We have implemented two approaches; the first is a rigorous coupled-wave model, whereas the second involves an integral equation for the electric field in the cell solved using the method of moments. The second approach can handle very general geometries, although it might be perhaps not as accurate as the first. For this case, since the solution of many integral equations is very time consuming, we devised strategies to reduce the computation. Our implementation is made numerically efficient by using only very few design frequencies, and accurately approximating a given filter transfer function by a quotient of polynomials which are functions of frequency. Additionally, the problem impedance matrices are conveniently represented as the product of a material independent matrix and a vector of dielectric constants, thus allowing us to fill the matrices only once. Our code has been parallelized for the Cray T3D, to take advantage of the parallel capabilities offered by PGAPACK. The main features of this genetic algorithm library are discussed, and simple example programs illustrating its use are presented. The results from a study of the solution convergence and properties as a function of some of the key parameters of the genetic algorithm are discussed. Novel solutions are illustrated for a very narrowband single grating transmission filter, and a relatively broadband double grating reflection filter at microwave frequencies.

I.INTRODUCTION

I.1. Dielectric grating filters

It has been demonstrated that planar dielectric layers combined with diffraction gratings exhibit sharp resonances due to the coupling of exterior evanescent diffractive fields to the leaky modes of dielectric waveguides. With proper choice of the cell size efficient switching of energy between (nearly) totally reflected zero-order mode and (nearly) totally transmitted zero-order mode is achieved over a small variation in wavelength. Such property of frequency selectivity leads to the possibility of filter designs whose arbitrarily narrow linewidths can be controlled by the choice of modulation amplitude and mode confinement. In optics the guided-mode resonance effect has been combined with classical antireflection properties of thin film structures, leading to the design of symmetric reflection filters, with low sidebands over wavelength ranges related to the number of films used and their dielectric constants [1]. It is known that the sideband reflection can be further decreased and extended over a larger spectral range by adding more homogeneous layers with dielectric constants and thicknesses determined by anti-reflection conditions. The position of the resonance is determined primarily by the grating period, and the linewidth of the filter can be controlled by the modulation of the grating. Recently, a combination of guided-mode resonance and high-reflection layer design has been demontrated to yield transmission bandpass filters [2]. A reflection filter of this type has also been demonstrated in the microwave region [3]. This work explores the possibility of finding new types of waveguide-grating filter solutions in the optical as well as the microwave regions, by investigating different geometries and materials for gratings and layers. This approach might lead to alternative or improved design solutions.

In the above studies 'forward scattering' analysis techniques were used to obtain the filter response, starting from a set of user specified filter parameters and relying on known properties of anti-reflection (half-wavelength) or high-reflection (quarter wavelength) homogeneous layers/gratings for reflection and transmission filters, respectively. By contrast, we start from a desired filter response and determine the grating/layer configuration and dielectric constants of the solution, without constraining the thicknesses to be a specified fraction of the resonance wavelength. This approach is particularly appealing in the microwave region where extending the approach traditionally used in optics would yield solutions which are too thick for practical use. At the same time the potential reduction in size can be beneficial in optics too, since thinner filters have a wider range of application.

The particular type of design described here can be thought of as an inverse-source problem, since it entails finding a distribution of sources which produce fields (or quantities derived from them) of given characteristics. Electromagnetic sources (electric and magnetic current densities) in a volume are related to the outside fields by a well known linear integral equation. Additionally, the sources are related to the fields inside the volume by a constitutive equation, involving the material properties. Then, the relationship linking the fields outside the source region to those inside is non-linear, in terms of material properties such as permittivity, permeability and conductivity. In our approach the solution of the non-linear inverse problem is cast as a combination of two linear steps, by explicitly introducing the electromagnetic sources in the computational volume as a set of unknowns in addition to the material and geometry unknowns. This allows to solve for material parameters and electric fields in the source volume which are consistent with Maxwell's equations. In order to reduce the number of unknowns and the complexity of the operators, we first invert for the permittivity and geometry only in the minimization of a cost function and then, given the materials and the geometry, we find the corresponding electric fields through forward scattering calculations. The sources thus computed are used to generate the far fields and the synthesized filter response.

I.2. Motivation for use of Genetic Algorithms

Some important issues can be explored with an inverse approach. The first relates to the shape of the resonance response of the filter; in particular, we have investigated the possibility of synthesizing a transmission response described by a Lorentzian lineshape with controllable linewidth, to achieve a very narrow band. On the other hand, we have also searched for solutions which have broadbanded response and sharp cutoff, to demonstrate that are also obtainable with these type of structures. A second issue concerns the actual number and configuration of gratings/layers required given a specific design curve; specifically, we have been looking for the 'simplest' solutions, i.e. those which involve the smallest number of gratings. In particular, we have also investigated the possibility of using more than two materials to make one grating, to achieve symmetric responses, exhibiting a sharp resonance and low side-bands. We have seen that there is a trade-off between the values of the dielectric constants (resulting in the modulation) and the thickness of the periodic cell. Since it is our interest to examine the possible existence of designs which might not be known, we have not restricted the choice of dielectric constants to a small set of familiar values, but instead have considered all the materials that can in principle exist. The above requirements translate in the necessity to sample the parameter space globally, rather than proceeding from a given starting point solution and effecting only local changes. For this reason we have chosen to perform the inversion process using a genetic algorithm.

Genetic algorithms (GAs) were developed by John Holland [4] and are based on an analogy with natural selection and population genetics. The past several years have seen increasing interest in using GAs to find solutions to difficult optimization problems in a variety of application areas, and GAs have proven to be a successful and robust technique. There are several reasons for this. First, GAs make no assumptions about the function to be optimized, their only requirement being a performance measure, some form of problem representation, and operators that generate new population members. Second, GAs are a global optimization method, as opposed to a local gradient-based technique, and look for solutions in a broad range of the parameter space. GAs and gradient-based optimization methods have complementary strengths and weaknesses; a hybrid algorithm of both methods can often outperform either one individually. Finally, GAs are a naturally parallel algorithm and can readily take advantage of the computational power of modern parallel computers.

As opposed to other optimization methods, GAs produce a population of candidate solutions instead of just a single solution, where the problem unknowns have been represented by a string of (encoded) numbers. Then GAs work by assigning a merit value to each string in the population according to a problem-specific fitness function. A 'survival-of-the fittest' step selects strings from the current population, according to their fitness. These strings recombine using operators such as crossover or mutation to produce a new generation of strings that are more fit than the previous ones, thereby evolving the solution towards a global extremum. The parallel genetic algorithm package PGAPACK [5] has been used in our work. Key features of PGAPACK are discussed in Section II with the objective of giving the reader an appreciation for the library's full range of capabilities, and not just the specific aspects utilized in this study.

I.3. Application of GAs to design problems in optics

A variety of optimization problems employing genetic algorithms have been reported in the literature in optical signal and image processing, diffractive optics, thin-film optics, image formation and tomography [6-20]. Genetic algorithms have been utilized for optimization of the quantized phase or amplitude in each element of a two-dimensional array to achieve the desired far-field intensity distribution. These display devices can find practical applications in optical information processing, optical pattern recognition, optical interconnections and spatial filtering. Yoshikawa et al. [6] have generated a 64x64 element array with 16 phase levels that reconstructs a desired simple image upon illumination with a uniform, monochromatic light source. The design of a spatial filter with a binary amplitude distribution used in a joint-transform optical correlator to discriminate between two given images has been reported by Mahlab et al. [7]. Takaki et al. [8] improved the pattern discrimination ability in an optical filtering system employing a liquid-crystal active lens by utilizing a genetic algorithm to find the optimum 8-level phase distribution in a 64x64 pixel array. An optimization of the spatial amplitude filter in a generalized optical Fourier transform processor for a simple pattern recognition task was presented in [9]. The problem of image deconvolution utilizing a microgenetic algorithm was addressed by Johnson and Abushagur [10]. Numerical examples illustrated the reconstruction of two binary images from their convolution without any prior knowledge about the two distributions except their regions of support.

The design of diffractive optical elements may require the encoding of large non-periodic phase arrays that would involve a large chromosome chain in the genetic algorithm thus becoming computationally expensive. To overcome this problem Brown and Kathman [11] reduced the number of variables by encoding the phase function of the diffractive element by its Fourier coefficients and used a genetic algorithm with floating point variables and variable mutation variance to design single and multiple, cascaded diffractive elements for laser beam shaping.

An optical implementation of a genetic algorithm was demonstrated by Friedman et al. [12] in a hybrid electro-optical system that exploits the inherent parallelism of optical architectures. The computational speed can thus be increased by allowing the vector-matrix multiplications and summations of binary elements to be performed optically while the rest of the algorithm is executed by a digital computer.

Promising results for application of genetic algorithms to tomography to achieve increased accuracy over conventional reconstruction methods for reduced number of projections have been reported recently by Kihm and Lyons [13,14]. To improve the convergence and reduce the computation time for the reconstruction of the unknown field from its measured projections, a hybrid genetic algorithm was devised where each new generation is partially created by the conventional genetic algorithm operators and partially by a concurrent Simplex operator [14].

Genetic algorithms have been proposed by E. Betensky [15] as useful tools in optimization of gaussian optics systems for improved lens design. To minimize the merit function, the algorithm selects the best set of structural changes to be applied to a starting design from a predefined list of permissible operators. X. Chen and K. Yamamoto [16] have designed a camera lens with seven elements by encoding the physical parameters of each lens in a chromosome. The authors found the definition of the merit function to be more important in convergence towards an optimum solution rather than the evolution strategy itself and used a two level merit function that includes the desired lens specifications (focal lengths, f-number and field of view). In a non-imaging application of genetic algorithms, I. Ashdown [17] has discussed the optimization of the near-field and far-field photometric distribution for the design of direct and indirect illuminating systems.

The design of diffractive optical elements using a genetic [18] and microgenetic algorithm [19] was reported by Johnson et al. Fan-out gratings for optical interconnection applications have been obtained with the desired intensity in each diffracted order by encoding the phase of each cell of a NxN matrix of cells and using fast Fourier transform to calculate the diffraction pattern. A rigorous coupled mode analysis was employed to compute diffraction efficiencies for binary periodic structures with subwavelength feature sizes and multiple layers. These corrugated dielectric gratings were optimized with the microgenetic algorithm to achieve reflection and transmission characteristics of various diffractive optical elements such as polarizing beamsplitters, antireflection coatings, resonant reflectors and fan-out gratings. The resonance reflector generated with the microgenetic algorithm was found by optimizing a dual-surface corrugated grating to yield 100% reflectance at one given wavelength. This subwavelength structure can form the basis of narrow-band reflection [1,20] and transmission filters [2,21] exploiting the guided-mode resonance effect. However, to obtain desired filter characteristics with specified linewidth, high efficiency, symmetrical lineshape and low sidebands, the optimization process must be performed for a set of wavelength points spanning the entire spectral range of interest and allowing the algorithm to search in a wider parameter space to find the period and fill factor of the grating and the optimum thicknesses and refractive indices of a multilayer structure.

Michielssen et al. [22] have designed practical lowpass and highpass optical filters using a real-encoded genetic algorithm. The dielectric constants and the number of homogeneous layers were fixed allowing the genetic algorithm to optimize the thicknesses of the layers to obtain the specified response. This genetic algorithm implementation can be applied to design other types of optical components, however, the use of only homogeneous layers restricts the range of devices that can be designed. Furthermore, structures containing subwavelength gratings as well as homogeneous layers can produce optical components such as bandpass filters, antireflection surfaces, wave plates, and polarization-selective mirrors with significantly fewer layers than the corresponding devices that employ homogeneous layers only.

II. PGAPACK GENETIC ALGORITHM LIBRARY

PGAPACK is a (parallel) genetic algorithm library that provides most capabilities desired in a genetic algorithm, in an integrated, seamless, and portable manner. It supports multiple data types, Fortran and C interfaces, a simple interface for novice and application users, multiple levels of access for expert users, object-oriented design, extensibility, multiple GA operators, and parallel portability across uniprocessors, multiprocessors, and workstation networks. Options allow the user to specify the population size, stopping criteria, and many additional features including whether duplicate strings should be allowed in the population, and whether to mutate or crossover strings, or to mutate and crossover strings. Furthermore, it supports native data types: binary-valued, integer-valued, real-valued, and character-valued strings. The user can customize the genetic algorithm by supplying function(s) to customize a particular operator(s) while still using one of the native data types. Additionally, the user can define a new data type, write the data-type-specific low-level GA functions (i.e., crossover, mutation, etc.), and have these functions executed by the high-level data-type-neutral functions. A data-hiding capability provides the full functionality of the library to the user, in a transparent manner, irrespective of the data type used.

In its simplest form, a program using PGAPACK can be written invoking only four library subroutines and a string evaluation function. Figure 1 shows a simple example program which maximizes the number of bits of value 1 in a string. PGACreate initializes

PGAPACK, by defining a context variable, ctx, which contains the identifying features of the optimization being performed and which is passed to the library subroutines called in the program. The user need specify only three parameters: the data type, the string length, and the direction of optimization. All other parameters have default values in this mode of usage. PGASetUp initializes all parameters and functions not explicitly set by the user to default values. The genetic algorithm 'machinery' is encapsulated within the single subroutine PGARun. Its second argument is the name of a user-provided function, 'evaluate', that will be called whenever a string evaluation is required. PGAGetBinaryAllele returns the value of the i-th bit of string p in population pop. PGADestroy releases all memory allocated by PGAPACK.

II.1. The merits of different types of encoding

An area of controversy in the GA literature is how to represent real-valued problem variables. There are two choices; a bit string encoding or an explicit real-valued representation. In a bit string representation, a subset of N bits in the string is used to encode each real value. Retrieving the encoded real value from a binary substring is then a two-step process. First, the substring is decoded into an unsigned integer $k \in [0, 2^N - 1]$

$$k = \sum_{i=1}^{N} b_i 2^{i-1}$$

Second, k is mapped to a real value r on the interval [L,U] according to

$$r = k \ x \ \frac{U - L}{2^N - 1} + L$$

There are several reasons for the popularity of the binary string representation. One reason is historical; the work of Holland and his collaborators popularized the binary representation. Additionally, binary strings are general enough to encode almost any data structure, are easy to manipulate and use straightforward operators (e.g., bit complement mutation). Furthermore, binary strings are easier to analyze theoretically than more complex representations, and theoretical arguments have been put forward that claim they are optimal in the sense of the number of patterns each bit can represent [23]. However, binary strings have several limitations. First, they require extra computational effort to decode into real values. Second, depending on the problem, decoded real values may not be accurate enough to represent the optimal value of the decision variable and a large number of bits may be necessary if a high-precision result is desired. However, as the number of bits increases, the size of the search space grows exponentially. Furthermore, hybrid algorithms that combine a GA with a gradient method for local improvement are often most desirable. Bit string representations, however, are uncommon in gradient methods, thus making the development of a hybrid algorithm more difficult. Finally, in a bit string representation two numbers that are contiguous in their decimal representations may be far from each other in their binary representations. For example, 3 and 4 are consecutive integers, yet their 3-bit binary representations, {011} and {100}, differ in the maximum number of bit positions (technically, this is known as a Hamming cliff). As a result, small changes to the string may result in large changes to the fitness values. An alternative to the standard binary encoding that maintains a contiguous representation is a *Gray code* encoding. Gray codes define an alternative different mapping of binary strings to integer values such that consecutive integers differ in only one bit position. PGAPACK provides functions that allow the user to encode or decode integer or real values using either a standard binary or reflected Gray code representation.

Rather than use a binary or Gray encoding, however, many prefer to use an explicit real-valued string. A number of empirical studies [24-28] favor using a real representation because of more accurate answers, greater consistency from run-to-run, and faster computational performance over those of a binary one. There are several advantages to a real encoding. First, unless many bits are used, real encodings have higher precision than binary encodings. Second, real encodings are computationally more efficient since no encoding and decoding needs to be done. Third, as a result of avoiding Hamming cliffs real-valued strings have the property that small changes in a variable's value lead to small changes in the function value. Fourth, real encodings allow for a larger range of (sometimes more natural) operators for crossover and mutation. Fifth, real values are more natural and intuitively closer to the problem space, since it is easier for a person to grasp the value of a real number than a string of bits. Finally, by using a representation that is in common use it is easier to create a hybrid of the GA with another optimization method.

II.2. Data Structure Neutrality and Custom Data Types

Several features of PGAPACK were developed specifically for application users who wanted a versatile, powerful, easy-to-use GA package. These features include defining custom data types, the ability to easily integrate hill-climbing heuristics within a GA, and a parallel implementation. PGAPACK is a data-structure-neutral library. By this we mean that a data-hiding capability provides the full functionality of the library transparently to the user, *irrespective of the data type* (coding) used. PGAPACK supports four native data types. The Bit data type (i.e., |1|0|1|1|) is the traditional GA coding. Here it is implemented by using each distinct bit in a computer word as a gene, making the software very memory-efficient. The Integer data type (i.e., |3|9|2|4|) supports strings that are integer valued. Such codings are often used in routing and scheduling problems. The Float data type (i.e., |4.2|7.1|6.3|0.8|) is often used in numerical optimization applications. The Character string data type (i.e., |h|e|l|l|o|) is useful in symbolic applications.

The use of a *context variable* (ctx) allows the choice of data type to be hidden. The context variable is a pointer to a C language structure which is itself a collection of other structures. These (sub)structures contain all the information necessary for a GA run such as data type, parameter values, which functions to call, operating system parameters, debugging flags, initialization choices, and internal scratch arrays. By 'hiding' the data type selected and specific functions that operate on that data type in the context variable, the high-level functions can be called independently of the data type selected.

PGAPACK may be extended in two ways. First, the user may replace some of the standard functions with user-defined functions for use with one of the native data types. Second, it may be extended by defining a new data type. To create a new data type the user must write data-structure-specific functions to allocate memory, perform mutation, and perform crossover. We illustrate the definition of a new data type with an example, aimed at showing the basic steps involved. Consider designing a composite material where the goal is to minimize the weight subject to satisfying certain electromagnetic and geometric constraints. The choices to be made are the type of material to use in each layer of the composite and the thickness of the material. The choice of material can be represented by an integer pointing to an entry in a data-base of electric permittivities/magnetic permeabilities and the thickness of the material by a real value. Such a formulation would use a string of length twice the number of layers. Half the genes would be real and the other half integer.

Figure 2 contains the main program for this example. NUM_LAYERS and NUM_MATERIALS are the number of layers in the composite material and the number of material choices for each layer, respectively. The structure *composite* defines the new data type. This may be interpreted as a logical string consisting of an integer part, *layer*, that specifies the material type, and a real part, *thick*, that specifies the thickness of the selected material. Next, function prototypes for the user's objective function (*weight*), the evaluation function (which plays the role of a wrapper in this example, as will become clear in the following), and the mutation appear. For the sake of brevity only the mutation function is outlined here, but a complete program would require definitions for several other functions [5]. Note the use of PGA_DATATYPE_USER in the PGACreate call to specify that a custom data type will be used and the call to PGASetUserFunction to specify the user's mutation function (CompMutation).

Figure 3 contains the user's mutation function *CompMutation* for the composite data type. Each gene has a probability *mr* of being changed. If a mutation to a real gene occurs, a Gaussian random variable is added to the current real value. If a mutation to an integer variable occurs, the existing material choice is replaced with one that is randomly selected. When using custom data types it is not possible to maintain the (p,pop) abstraction to specify an individual (a string and associated fields). Instead, PGAGetIndividual(ctx, p, pop) returns a pointer to the individual (the string and associated fields) specified by (p,pop). One field of this structure is *chrom*, the string itself.

Figure 4 contains the evaluation function. The scratch arrays *l* and *t* are loaded with the integer and real values, respectively. These are then passed to *weight* , the user's true evaluation function. This type of usage allows *weight* to be called with no changes required by the specific encoding.

II.3. Hybrid Genetic Algorithms

Many successful GA applications use problem-specific information contained in either the solution encoding, or in problem-specific operators, or in a hill-climbing routine. The last usage, combining a GA with a hill-climber, is often referred to as a *hybrid GA* and has a broad use. Hill-climbing heuristics attempt to improve a solution by moving to a *neighboring* solution of smaller residual value. Whenever the neighboring solution is better than the current solution, it replaces the current solution.

Genetic algorithms and hill-climbing heuristics have complementary strengths and weaknesses. GAs are good at finding promising areas of the search space, but not as good at fine-tuning within those areas. Hill-climbing heuristics, on the other hand, are good at fine-tuning, but lack a global perspective. Practice has shown that a *hybrid* algorithm that combines GAs with hill-climbing heuristics often results in an algorithm that can outperform either one individually.

There are two general schemes for creating hybrid algorithms. The simplest is to run the genetic algorithm until it terminates and then apply a hill-climbing heuristic to each (or just the best) string. In this case, the hill-climbing routine starts with 'good' solutions found by the GA. The second approach is to integrate a hill-climbing heuristic with the genetic algorithm. PGAPACK supports hybridization in several ways. First, by passing the context variable as a parameter, the hill-climbing function has access to solution and parameter values, debug flags, and other information. Second, PGAPACK functions allow the hill-climbing function to read and write allele values, to read integer or real numbers encoded as binary or Gray code strings, and to encode integer or real numbers as binary or Gray code strings. Third, other PGAPACK library functions allow the user's hill-climbing routine to read and write evaluation function values, and to get and set the flag that indicates whether a string evaluation is up to date.

II.4. Parallel Implementation of PGAPACK

One way to parallelize a GA is to execute in parallel the loop iterations that create generation t+1 from generation t. Most steps in this loop, such as crossover, mutation, evaluation, can be executed in parallel. The execution efficiency, however, depends upon the computer architecture and parallel execution overhead, the number of new population members created at each generation (the degree of parallelism), and the computational cost of the steps being executed in parallel (the granularity). However, in many real-world applications the dominant computational cost is the time to execute function evaluations and the performance benefits from parallel execution may be achieved by parallelizing only this step. The parallel implementation in PGAPACK uses a master/slave algorithm in which one processor, the *master*, executes all steps of the genetic algorithm except the function evaluations, which are executed by *slave* processors. Note that since we use a message-passing programming model to implement the master/slave algorithm, there may be a significant parallel execution overhead. Focusing only on parallelizing the function evaluations allows for modest data distribution requirements (just the strings to be evaluated) and minimal synchronization requirements.

There are two primary considerations in determining the performance advantage of using the master/slave model. First, the speedup will vary according to the amount of computation associated with a function evaluation and the computational overhead of distributing and collecting information to and from the slave processors. Second, the number of function evaluations that can be executed in parallel will limit the speedup. This number depends on the population size and the number of new strings created at each generation. In a generational replacement model, the entire population may be evaluated in parallel. In the more popular steady-state model, however, a number of new strings are produced as a user defined percentage of the population, and the degree of parallelism is minimal. By default PGAPACK replaces 10% of the population. In our experience this percentage usually provides an acceptable degree of parallelism, while retaining the superior performance characteristics of the steady-state model.

PGAPACK is implemented using the message-passing interface (MPI) standard [29]. MPI is a *specification protocol* of a message-passing library for parallel computers and workstation networks---it defines a set of functions and their behavior. Implementations of MPI exist for both sequential (uniprocessors) and parallel (multiprocessors, multicomputers, and workstation networks) computer hardware, thereby allowing PGAPACK to run on all these machines without any code changes.

III. DIELECTRIC GRATING FILTER DESIGN

We discuss an inverse approach to designing dielectric filters realized as stacks of inhomogeneous gratings and (possibly) homogeneous layers of materials, as described in Fig. 5; in the last few years such devices have been demonstrated in optical technology, but they are not common at microwave frequencies. The problem amounts to finding the stack's geometric configuration and permittivity values which correspond to a specified reflectivity/transmittivity response. We solve for geometry and materials using a customized version of PGAPACK to minimize a cost function obtained by calculating the deviation between the synthesized value of reflectivity/transmittivity and the desired one. For each candidate solution we calculate the correspondent reflectivity/transmittivity by a forward scattering method; specifically, we evaluate the candidate's fields or currents consistently with Maxwell's equations. These quantities act as secondary unknowns, whose solution is used to obtain the reflection/transmission efficiencies and eventually the cost function. In the following we outline two techniques that have been used for this purpose.

The first technique is a rigorous coupled-wave analysis [30,31], a differential method for obtaining exact solutions of Maxwell's equations for diffraction of electromagnetic waves by periodic structures. The method is accurate, efficient and stable and has been applied to a wide variety of gratings (holographic, binary single or multilayer, surface relief, dielectric or metallic, anisotropic) for arbitrary polarization and angle of the incident beam. Because the accuracy of this scattering model is well established, we used this approach to study issues concerning convergence and stability of the GA solutions as a function of the different types of encoding and other key GA parameters, for a single grating reflection filter.

The second technique consists of numerically solving the integral equation for the electric field in the periodic cell using the method of moments [32]. While this second approach might perhaps not be as accurate, it is more flexible in its ability to model diverse geometries, even completely different from that of Fig. 5. Indeed only few aspects of the formulation are specific to the design problem at hand and the reader will find that the general methodology is applicable to other EM design problems. Since the solution of many integral equations (one for each frequency/illumination angle) is computationally expensive, we devised strategies to reduce the required time. Although the impedance matrix depends on the solution vector of materials and boundaries candidates, it can be formed as a product of a solution-independent matrix and a vector. This procedure allows us to fill the set of frequency-dependent impedance matrices only once. Additionally, the number of design frequencies at which the integral equations are actually solved is a small set of values within the frequency range of the desired filter response. The reduction is afforded by approximating the desired filter response by a quotient of frequency dependent polynomials, through the procedure of transfer function parameter estimation described in [33]. A set of design frequencies is chosen suffient to find the unknown polynomial coefficients. Furthermore, full advantage has been taken of the parallel implementation of PGAPACK for the Cray T3D. The parallelization scheme used for the genetic algorithm is an intuitive, simple master-slave configuration, where the expensive evaluation cycles are distributed among the processors. A complete description of this approach is presented in [34].

III.1. Forward Scattering via Rigorous Coupled - Wave Technique

In the present study the formalism of [30,31] is applied to calculate diffraction efficiencies of a multilayer structure consisting of binary gratings with rectangular dielectric constant profile and homogeneous layers for a TE-polarized incident plane wave as illustrated in Fig. 5. In the grating layers the periodic modulation of the dielectric constant can be represented as a Fourier expansion

$$\varepsilon(x) = \sum_{h=-\infty}^{\infty} \varepsilon_h \exp[jhKx] \tag{1}$$

where K = $2\pi/\Lambda$, Λ is the grating period and ε_h are the coefficients of the Fourier expansion.

In the rigorous coupled-wave approach the tangential electric (E_y) and magnetic (H_x) fields inside a grating layer are expanded in terms of spatial (Floquet) harmonics

$$E_y(x,z) = \sum_{i=-\infty}^{\infty} S_{y,i}(z)\exp[-jk_{x,i}x]$$
$$H_x(x,z) = -j\left(\frac{\varepsilon_0}{\mu_0}\right)^{1/2} \sum_{i=-\infty}^{\infty} U_{x,i}(z)\exp[-jk_{x,i}x] \qquad (2)$$

where $k_{x,i}$ is the wavevector component along x given by the Floquet condition: $k_{x,i} = k_0 n_I \sin\theta - iK$, k_0 is the wavenumber in free space, ε_0 and μ_0 are the free-space permittivity and permeability, and $S_{y,i}$ (z) and $U_{x,i}$ (z) are the normalized amplitudes of the ith space-harmonic fields. These field expansions must satisfy Maxwell's equations assuming harmonic time dependence $e^{j\omega t}$ and uniform field along the y direction.

$$\frac{dE_y(x,z)}{dz} = j\omega\mu_0 H_x(x,z)$$
$$\frac{dH_x(x,z)}{dz} = j\omega\varepsilon_0\varepsilon(x)E_y(x,z) + \frac{j}{\omega\mu_0}\frac{d^2E_y(x,z)}{dx^2} \qquad (3)$$

Substituting the field expressions (2) and the dielectric constant expansion (1) in equations (3) and eliminating the amplitudes $U_{x,i}$, one obtains an infinite set of second-order coupled-wave equations given by

$$\frac{d^2S_{y,i}(z')}{d(z')^2} = \frac{k_{x,i}^2}{k_0^2}S_{y,i}(z') - \sum_{h=-\infty}^{\infty}\varepsilon_{i-h}S_{y,h}(z') \qquad (4)$$

To obtain a numerical solution, the infinite number of coupled-wave equations are truncated to a finite number N of harmonics and the eigenvectors and eigenvalues of the resulting NxN system matrix are calculated. The solution to the system of second-order coupled-wave equations can be written as

$$S_{y,i}(z) = \sum_{m=1}^{N} W_{i,m}\left[C_m^+\exp(-k_0q_mz) + C_m^-\exp(k_0q_mz)\right] \qquad (5)$$

where q_m is the positive root of the m-th eigenvalue, $W_{i,m}$ is the corresponding eigenvector, while C_m^+ and C_m^- are unknown amplitude constants to be determined from the boundary conditions. The tangential electric and magnetic fields E_y and H_x in the grating region can then be obtained from Eq. (2) and Maxwell's equations (3). In the homogeneous layers the tangential electric and magnetic fields are written as a sum of forward and backward propagating plane waves with unknown constant field amplitudes.

In the input (medium I) and output region (medium III), the electric field can be expressed in terms of the incident and diffracted propagating waves in the form

$$E_{y,I} = \exp(-jk_{z,0}z) + \sum_{i=-(N-1)/2}^{(N-1)/2} R_i \exp(jk_{z,I,i}z) \exp(-jk_{x,i}x)$$

$$E_{y,III} = \sum_{i=-(N-1)/2}^{(N-1)/2} T_i \exp(-jk_{z,III,i}(z-D)) \exp(-jk_{x,i}x)$$

where R_i and T_i are the normalized electric field amplitudes or the reflected and transmitted diffraction orders, respectively, and D is the thickness of the structure. The wavevector components along x and z are determined from phase matching requirements and imposing the Floquet condition $k_{x,i} = k_0[n_I \sin\theta - i(\lambda/\Lambda)]$, $k_{z,p,i} = (k_p^2 - k_{x,i}^2)^{1/2}$ where p = I or III.

Imposing the boundary conditions at the interfaces between the L adjacent layers and between the structure and the input and output media, a system of 2N x (L+1) equations is obtained with 2N x L unknown field amplitudes in the layers, N unknown reflected wave amplitudes R_i and N unknown transmitted wave amplitudes T_i. This system can be solved directly to find all unknowns simultaneously but this would be highly inefficient, particularly for large L. To increase the efficiency of the numerical computation the number of equations can be reduced by eliminating the 2N x L unknown amplitudes in the layers (C_m^+ and C_m^-) resulting in a system of 2N equations for unknowns R_i and T_i. To prevent numerical instabilities associated with inversion of ill-conditioned matrices and increase the accuracy of the computation, the enhanced transmittance approach presented in [31] is utilized to calculate the reflected and transmitted wave amplitudes. Once the reflected and transmitted diffraction orders have been found, the diffraction efficiencies are determined from the formulas

$$DE_{R,i} = \mathrm{Re}\left[\frac{k_{I,z,i}}{k_{I,z,0}}\right]|R_i|^2$$
$$DE_{T,i} = \mathrm{Re}\left[\frac{k_{III,z,i}}{k_{III,z,0}}\right]|T_i|^2 \qquad (6)$$

III.2. Forward Scattering via E-field Integral Equation

The following formulation is also presented in a more detailed fashion in [34]. It is suitable for geometries such as the dielectric grating stack illustrated in Fig. 5, but can handle more general structures including, for example, those having a thickness *t(x)* variable over the cell. The cell material is characterized by possibly complex permittivity and permeability. In analogy with the alternative formulation presented in the previous section, the polarization with the electric field parallel to the strip will be considered (TE - polarization). To properly pose the scattering problem for a periodic structure, the excitation field must be a function with constant amplitude and linear phase. The incident field is defined as [32]

$$E^i(x,z) = E_0 \psi_0(x)\ e^{jk_{z0}z} \qquad (7)$$

where the Floquet Harmonic is given by

$$\psi_m(x) = \frac{1}{\sqrt{\Lambda}} e^{-jk_{x_m}x} \qquad (8)$$

and

$$k_{x_m} = \frac{2\pi}{T_x} m + k_{x_0}, \quad k_{x_0} = k_o \sin\theta^i$$

$$k_{z_m} = \begin{Bmatrix} \sqrt{k_o^2 - k_{x_m}^2}, & k_o \geq k_{x_m} \\ -j\sqrt{k_{x_m}^2 - k_o^2}, & k_o \leq k_{x_m} \end{Bmatrix}$$

and the $e^{j\omega t}$ time convention is again used. According to the electric field integral equation, the unknown induced current J(ρ) is found, at each design frequency/illumination angle, from

$$E^i(\rho) = \frac{1}{j\omega\varepsilon_0 \chi(\rho)} J(\rho) - E^s(\rho) \tag{9}$$

where all components are y directed, and χ is the contrast function $\chi(\rho) = \varepsilon_r(\rho) - 1$. The scattered field is found from integrating the induced currents over the grating

$$E^s\,(x,z) = \frac{-\omega}{4} \int_0^{T_x} \int_0^{t(x)} \mu(x',z') J(x',z') G_p(x,z|x',z') dx', dz' \tag{10}$$

where G_p is the two-dimensional periodic Green's function – the outgoing Hankel function of order zero – representing the field due to source points within each cell [33]. Using this periodic spatial Green's function, the integration area is that of one periodic cell. Equation (10) contains an integrable singularity, occurring as the source and observation points are made to coincide. A method for isolating this singular point and performing the integration in an efficient manner is summarized in [34, 35]. The result is

$$E^s\ (x,z)=\int_0^{T_x} dx'\int_0^{t} dz' J(x',z')\left[Z_m^P(x,z|x',z')-Z_m^P(x,z|x',z'+\delta)\right]$$

$$+\sum_m \tilde{Z}_m \tilde{I}_m^{\pm}\psi_m(x)\ e^{\mp k_{z_m}(z+\delta)} \qquad (11)$$

The method of moments is used to solve the above integral equation. The numerical solution of (9) is found by first discretizing the current over a periodic cell in a pulse basis set

$$J(x,z)\ =\sum_{p'=1}^{P}\ \sum_{q'=1}^{Q} C_{p'q'}\pi_{p'}(x)\pi_{q'}(z),\quad P'x\ Q'=P'Q' \qquad (12)$$

and subsequently using point matching, with the testing functions being delta functions. The matrix system for the unknown coefficients C is then

$$\left\langle E^i,T_{pq}\right\rangle=\sum_{p'q'} C_{p'q'}\langle\ \frac{1}{j\omega\varepsilon_0\chi}\pi_{p'}\pi_{q'},T_{pq}\ \rangle$$

$$-\ \sum_{p'q'}\left[\left\langle E^{ss}(x,z),T_{pq}\right\rangle+\left\langle E^{sn}(x,z),T_{pq}\right\rangle\right]\quad pq=1,2,\dots P'Q' \qquad (13)$$

The matrix of (13) depends on the materials through the contrast χ. However, it can be represented as a product of a material-independent matrix times the vector of the contrasts at each discretization cell. Therefore, the basic matrix is filled only once for each design frequency, and it is simply updated at each iteration step by multiplication with the current vector of contrasts. For a cell with general inhomogeneities there is one independent contrast value for each discretization cell used in our method of moments. For the type of devices of concern to us, we constrain the grating periodic cell to be composed of a number of regions with different materials, separated by boundaries, and arranged in a combination of homogeneous horizontal layers and/or strips with χ varying along the x and z directions. Alternative geometries, such as those illustrated in Fig. 6, can also be modeled with this scheme. Then the set of inversion parameters for the GA, i.e. the different values for χ and boundary locations along x and z, is much smaller than the number of discretization cells. A mapping algorithm transforms this reduced parameter set for a candidate solution to the full vector of contrasts at each discretization cell.

Assuming that only the dominant (m = 0) mode is propagating, the reflection and transmission coefficients are found from evaluating the total field at z >> t, and z <<0 respectively,

$$R = \frac{-\omega}{4E_0}\frac{1}{2jk_{z0}}\int_0^t dz'\mu\tilde{J}_0(z')\ e^{jk_{z0}z'} \tag{14a}$$

$$T = 1 - \frac{\omega}{4E_0}\frac{1}{2jk_{z0}}\int_0^t dz'\mu\tilde{J}_0(z')\,e^{-jk_{z0}z'} \tag{14b}$$

where the transform of the current is given in [33].

In designing a filter the desired behavior of reflection or transmission efficiencies *(RR** or *TT*)* within a continuous range of frequencies is often specified; since the coefficients of the current are frequency dependent it would appear that to evaluate the response of a candidate many method of moments solutions of (9) must be calculated. However, any prescribed filter response must satisfy the condition of realizability. It is well known from classical lumped parameter filter design that realizability requires that the insertion loss, as a function of frequency, be representable as the ratio of two polynomials of even powers of frequency. Then the problem is reduced to representing the desired filter response by a quotient of polynomials, working with a set of prescribed values sufficient to determine the unknown coefficients. As discussed in [33], the general representation for a magnitude-squared network transfer function with poles and zeroes is given by

$$|F(\omega_i)|^2 = \frac{\sum_{j=0}^{N-1} B_j(\omega_i)^{2j}}{\sum_{j=0}^{N} A_j(\omega_i)^{2j}} + \sum_{j=0}^{K} C_j(\omega_i)^{2j} \tag{15}$$

where the quantity on the left-hand-side is either *RR** or *TT**. Equation (15) can easily be turned into a system of equations in the 2xN+K unknowns A_j, B_j and C_j. Hence the solution of (9) is calculated only at 2xN+K design frequencies, and the correspondent values for (14a) or (14b) are used to solve for (15). The polynomial approximation of (14a) or (14b) are then obtained at all frequencies of interest, and used in the evaluation of the residual. In our implementation we investigated prescribed insertion loss functions which can be represented very well by Equation 15, and in the case of a Butterworth response discussed in the following, exactly. Naturally, the frequency response of a distributed system might not be consistent with (15), and a thorough investigation of its applicability is beyond the scope of this work. However, since we are concerned with filter designs in the neighbourhood of one resonance, it is expected that (15) will apply approximately to the filter transmittivity and/or reflectivity. Indeed, for the examples reported in IV.2, we verified *a posteriori* that the solutions calculated by extrapolation from (15) were in agreement with those calculated directly.

IV. NUMERICAL RESULTS

IV.1. GA Convergence Properties

The results reported in this subsection were obtained using the forward scattering model outlined in Sec. III.1, together with a sequential implementation of PGAPACK. The objective was to study the convergence properties and the solution features of the GA as a function of some of its key parameters such as type of encoding (real versus binary), replacement percentages, and mutation probability for the same problem. To this end we generated N = 50 design values of reflectance, non-uniformly distributed in the range 0.546 nm - 0.554 nm of wavelengths, corresponding to a single grating filter of thickness d = 0.134 nm, period $\Lambda = 0.314$ nm, fill factor f = 0.5 and dielectric indices of refraction $n_H = 2.1$ and $n_L = 2.0$. This structure has a resonance at 0.5503 nm with reflectivity 0.9917. The angle of incidence was taken to be 0°, the medium of the incoming plane wave was assumed to be air, the medium of the exiting plane wave had refractive index 1.52, and 7 harmonic waves were used in the truncated series of Eq. 5. The residual was calculated as follows

$$\mathrm{Re}\,s = \left[\frac{1}{N} \sum_{i=1}^{N} \left| R_{ref}(i) - R_{cal}(i) \right|^{n} \right]^{1/n}$$

and the difference is taken between the reflectance value correspondent to the candidate GA solution (*cal*) and the reference. The value of *n* correspondent to the results reported in this subsection was 2. Keeping the cell periodicity fixed, the GA optimized for the thickness, fill factor and materials, choosing from a table of 13 values of user specified refractive indices. Of the many GA parameters we found that our problem was very sensitive to two, the mutation probability and the population replacement, when contrasting real versus binary encoding performance. Hence, the population size was fixed at 500, the crossover type was uniform with probability fixed at 0.8, and the number of generations and mutation probability were varied. For the real encoding the mutation type was chosen to be gaussian with standard deviation $\sigma = 0.1$. We specified to have the newly created strings undergo both crossover and mutation. The results illustrated in Figs. 7-9 are for real-encoding.

Figure 7 illustrates the behavior of the (smallest) residual as a function of the mutation probability after 200 generations, when replacing 200 strings per generation. The improvement with the increase in mutation probability is to be expected since, for real-encoded chromosomes, mutation plays a critical role in fine-tuning the solution. The fact that the residual decrease is not monotonic with increase of mutation probability is attributed to the limited statistical sample of realizations considered in the numerical tests.

Figure 8 shows the behavior of the residual as a function of time (one point every five generations, starting with the fifth itself) for various values of mutation probability. The same GA parameters as discussed above were retained. Note that again, the convergence is not monotonic, as shown in Fig. 8b, and that it is questionable to decide whether the probability 1.0 is preferable to 0.5. In fact, the residual drops faster in the case when the latter value is chosen. Again, it is possible that this effect is due to the limited statistical sample used in the numerical tests.

Figure 9 reports the behavior of the residual as a function of the number of strings replaced at each generation. As expected, at first the rate of convergence is higher when increasing the number of strings replaced at each generation, but only up to a point. In fact, the curve shows that replacing up to 50% of the population at each generation results in a small convergence rate, but at the same time the best rate was achieved when replacing 40%. Even accounting for the possible statistical limitation, the results shown nevertheless indicate that relatively high replacement percentages produce the best convergence.

Similar convergence tests were conducted using binary encoding with 10 bits to represent the thickness, 8 bits to represent the fill factor and 4 bits to represent the pointer to the set of materials. The same population number of 500 was retained, uniform crossover with probability 0.8, and both crossover and mutation were specified to occur in the formation of new strings. On the other hand, the number of generations was increased to 400, since the values of the residuals tended to remain high. Figures 10-12 illustrate the results for convergence.

Figure 10 illustrates the behavior of the (smallest) residual as a function of the mutation probability after 400 generations, when replacing 50 strings per generation. In contrast with the choice for real encoding, the smaller replacement value (10%) was chosen here, as recommended in the literature. Note the significant difference with the real encoding case in the impact of mutation probability on convergence; as suggested in the literature small values give the best performance, although the tests show non monotonic convergence, and higher sensitivity on mutation probability.

Figure 11 illustrates the convergence as a function of time, and it should be contrasted with Fig. 8 for the real case. We can see that for binary encoding, even after 400 generations, the residual is not as small as that obtained with real encoding after 200 generations. We attribute this fact to the limitations of the binary representation (although a large number of bits was used), or equivalently, the improved performance of the real encoding. Indeed we verified, through a few numerical tests, that by increasing the population size we could reduce the residual obtained for binary encoding to the smaller values obtained for the real case, at the cost of increased computation.

Figure 12 shows the convergence as a function of time corresponding to different replacement values. It is noted that, in contrast to the real encoding case, smaller differences in residual at convergence are obtained depending on the replacement choice, indicating that all four values perform equivalently, when accounting for different generations required to converge. In fact, in terms of computational cost, replacing 50 strings and running for 400 generations yields very similar results to replacing 100 (250) and running for 200 (80) generations. Hence binary encoding is less sensitive to values of replacement per generation and more sensitive to mutation probability.

Concerning the solution quality, our problem does not have a unique solution, at least for the limited range of values specified in the design and the current scheme for the residual calculations. The genetic algorithm identified several distinct solutions, all exibiting a resonance at the correct location, one of which was the structure originally used to generate the design data. The different solutions have different features, merits and drawbacks, and an evaluation of the solution quality is bound to be very problem specific. A preferred encoding has not been clearly identified by these tests.

IV.2. Novel Filter designs

The results reported in this subsection were obtained using the forward scattering approach outlined in Sec. III.2 together with a parallel implementation of PGAPACK. They are also illustrated in [34]. Additionally, we restricted our attention to real-encoded chromosomes, constructed as the sequence of material permittivities followed by the boundary locations. The order started with the bottom layer/grating and proceeded in the direction of the positive x direction. For real encoding the crossover operation implemented in PGAPACK is 'swap-only'; that is to say that when two parents combine to form an offspring, portions of their chromosomes are simply copied as such in the resulting string. By contrast other implementations exist where arithmetic or geometric averages of the allele values being swapped form the child strings [26]. In this portion of the work we always used uniform crossover and Gaussian mutation type. With this choice the allele is mutated by adding a random number obtained from a Gaussian distribution with zero mean and standard deviation prescribed by the user to be a percentage of the current allele's value. Typically, five to ten percent was specified in our test runs. Larger percentages resulted in less desirable (higher residual value) solutions for the same number of generations. One can see that, with this real encoding, crossover is the most important operation during the early generations, when the gross features of a good solution are evolved. Later on mutation becomes the critical mechanism which allows the fine features of the solution to emerge. In the following cases the crossover probability was fixed at 85% and we elected to have the recombining chromosomes always undergo crossover and mutation.

The essence of our parallelization strategy consists of distributing the work of evaluating the candidate solutions generated at each iteration among the 'slave' processors, while the 'master' receives the calculated residuals and proceeds to perform all the GA related operations. Hence, each 'slave' processor computes all the necessary MoM solutions for a given candidate. In order to do this efficiently each processor stores all the necessary impedance matrices, one for each frequency/illumination angle. This strategy is preferable to having the impedance matrices reside only on one processor and communicate with all others, as long as the matrices can all fit in the processors' memory. In this type of problem the order of the matrix is about 100, so that the storage of up to about 20 complex full matrices on one JPL Cray T3D processor (62 Mbytes) is possible. With the growth of the problem size one will eventually have to distribute the matrices among the processors. To keep a good load balance the number of processors has to be decided based on the number of replacement strings generated at each iteration. Ideally, each processor should do at least one set of MoM calculations per generation, to avoid idle time. However, if the number of generations is small and the overall calculations in the generational replacements is much smaller than the initial population, then it is most efficient to choose to number of processors based on the initial population size.

A novel design for a narrowline bandpass filter in the microwave region is presented in Figure. 13. By allowing the unknown dielectric constants to span the range between 1 and 10 - a realistic assumption in the microwave region - we have obtained a solution for a three-material single waveguide-grating transmission filter with a bandwidth of 0.7% of the central wavelength of 3 cm. The geometry of the cell and illumination condition is reported in the figure, together with the obtained filter response. The grating period Λ and thickness were fixed, and we solved for the two geometric boundaries (ranging between 0 and Λ) and three material permittivities (ranging between 1 and 10. The prescribed response is a Lorentzian line approximated according to (15); for this particular case we determined numerically that N=2, K=1 was sufficient to represent the Lorentzian with at least six digits of accuracy. As a results five design wavelengths were used to specify RR^*: 2.5, 2.75, 3, 3.25 and 3.5 cm. The residual was actually evaluated for a set of 103 wavelengths, not equally distributed in the range 2.5 - 3.5 cm, but rather having a denser distribution in a small region around the expected resonance (21 points in the range 2.98 - 3.02 cm). We took the population size to be 3000 with replacement through cross-over and mutation of up to 300 (steady state) at every generation and performed the calculations on the JPL Cray T3D. Gridding the cell with 21 x 7 points was sufficient to achieve convergence to the solution reported in the figure in about 150 iterations. For each iteration, the overall cost of replacement, i.e. the time necessary to evaluate the newly created strings at each iteration, was about 5 seconds using 64 processors.

As an example of a stopband filter, a response described by a fourth order Butterworth polynomial with bandwith of 8% of the center wavelength was input to the genetic algorithm as a prescribed reflectivity. A synthesized two-grating solution, the simplest realization with the smallest cell size which was found, is illustrated in Figure 14, with the obtained response contrasted with the desired one. Sixteen wavelengths were specified to exactly represent the Butterworth curve according to (10), and the residual was calculated at 51 points. Both sets of points were uniformly distributed in the range of interest of 2.5-3.5 cm. The trial cell size was chosen to be 2x2.7 cm and was gridded with 10x14 points. A population size of 4000 with 10% replacement was chosen, and convergence was reached in about 300 generations. The calculation was performed on the Cray T3D using 128 processors in about 1 hour with a cost of replacement of about 13 sec.

ACKNOLEDGEMENTS

We acknowledge the contributions of T. Cwik and C. Ditchman to the work described in [34], portions of which have been summarized here. Use of the Cray Supercomputers at the Jet Propulsion Laboratory, Pasadena, CA, is acknowledged. The research described here was performed at the Jet Propulsion Laboratory, under contract with the National Aeronautics and Space Administration.

REFERENCES

[1] S.S. Wang and R. Magnusson, "Multilayer waveguide-grating filters," *Applied Optics,* vol. 34, pp. 2414-2420, May 1995.

[2] R. Magnusson and S.S. Wang, "Transmission bandpass guided-mode resonance filters," *Applied Optics,* vol. 34, pp. 8106-8109, Dec. 1995.

[3] R. Magnusson, S.S. Wang, T.D. Black, and A. Sohn, " Resonance properties of dielectric waveguide gratings: theory and experiments at 4-18 GHz, " *IEEE Trans. Antennas and Propagation,* vol. 42, pp. 567-569, April 1994.

[4] J. Holland, "Adaption in Natural and Artificial Systems, "MIT Press, Cambridge, 1992.

[5] D. Levine, "Users guide to the PGAPACK parallel genetic algorithm library, " Technical Report ANL - 95/18, 1995. PGAPACK is available via anonymous ftp at *ftp.mcs.anl.gov* or fron URL *http://info.mcs.anl.gov/pub/pgapack/pgapack.tar.Z.*

[6] N. Yoshikawa, M. Itoh, and T. Yatagai, "Quantized phase optimization of two-dimensional Fourier kinoforms by a genetic algorithm," Opt. Lett. vol. 20, no. 7, pp. 752-754, April 1995.

[7] U. Mahlab, J. Shamir, and H. J. Caulfield, "Genetic algorithm for optical pattern recognition," Opt. Lett. vol. 16, no. 9, pp. 648-650, May 1991.

[8] Y. Takaki, K. Ishida, and H. Ohzu, "Incoherent pattern detection using a liquid-crystal active lens," Appl. Opt. vol. 35, no. 17, pp. 3134-3140, June 1996.

[9] H. Peng, H. J. Caulfield, J. Kinser, and J. Hereford, "Optimization filters design for GFT by genetic algorithm," Optical implementation of information processing, San Diego, California, July 10-11, 1995, Proc. Soc. Photo-Opt. Instrum. Eng., vol. 2565, pp. 74-84, 1995.

[10] E. G. Johnson and M. A. G. Abushagur, "Image deconvolution using a micro genetic algorithm," Opt. Comm. vol. 140, pp. 6-10, July 1997.

[11] D. Brown and A. Kathman, "Multi-element diffractive optical designs using evolutionary programming," Diffractive and holographic optics technology II, San Jose, California, February 9-10, 1995, Proc. Soc. Photo-Opt. Instrum. Eng., vol. 2404, pp. 17-27, 1995.

[12] M. Friedman, U. Mahlab, and J. Shamir, "Collective genetic algorithm for optimization and its electro-optic implementation," Appl. Opt. vol. 32, no. 23, pp. 4423-4429, August 1993.

[13] K. D. Kihm and D. P. Lyons, "Optical tomography using a genetic algorithm," Opt. Lett. vol. 21, no. 17, pp. 1327-1329, September 1996.

[14] D. P. Lyons and K. D. Kihm, "Tomographic-image reconstruction using a hybrid genetic algorithm," Opt. Lett. vol. 22, no. 12, pp. 847-849, June 1997.

[15] E. Betensky, "Postmodern lens design," Opt. Eng. vol. 32, no. 8, pp. 1750-1756, August 1993.

[16] X. Chen and K. Yamamoto, "Genetic algorithm and its application in lens design," Current developments in optical design and engineering VI, Denver, Colorado, August 5-7, 1996, Proc. Soc. Photo-Opt. Instrum. Eng., vol. 2863, pp. 216-221, 1996.

[17] I. Ashdown, "Non-imaging optics design using genetic algorithms," Journal of the Illuminating Engineering Society, vol. 23, no. 1, pp.12-21, 1994.

[18] E. G. Johnson, A. D. Kathman, D. H. Hochmut, A. Cook, D. R. Brown, and W Delaney, "Advantages of genetic algorithm optimization methodsin diffractive optic design," in Diffractive and Miniaturized Optics, S. H. Lee ed., Proc. Soc. Photo-Opt. Instrum. Eng., vol. CR49, pp. 54-74, 1993.

[19] E. G. Johnson and M. A. G. Abushagur, "Microgenetic-algorithm optimization methods applied to dielectric gratings," J. Opt. Soc. Am. A vol. 12, no. 5, pp. 1152-1160, May 1995.

[20] S. S. Wang and R. Magnusson, "Theory and applications of guided-mode resonance filters," Appl. Opt. vol. 32, no. 14, pp. 2606-2613, May 1993.

[21] S. Tibuleac and R. Magnusson, "Diffractive narrow-band transmission filters based on guided-mode resonance effects in thin-film multilayers," IEEE Phot. Tech. Lett vol. 9, no. 4, pp. 464-466, April 1997.

[22] E. Michielssen, S. Ranjithan, and R. Mittra, "Optimal multilayer filter design using real coded genetic algorithms," IEE Proceedings J. vol. 139, no. 6, pp. 413-420, December 1992.

[23] D. Goldberg, " Genetic algorithms in search, optimization and machine learning, " Addison-Wesley, Reading, MA, pp. 80-82, 1989.

[24] L. Davis, "Handbook of Genetic Algorithms, "Van Nostrand Reinhold, New York, pp.62-70, 1991.

[25] L. Eshelman and J. Schaffer, " Real-coded genetic algorithms and interval-schemata, " in Foundations of Genetic Algorithms 2, D. Whitley, Ed., Morgan Kaufman, San Mateo, CA, pp. 187-202, 1993.

[26] Z. Michalewicz, " Genetic algorithms + Data structures = Evolution Programs, " Springer-Verlag, New York, pp. 75-82, 1992.

[27] N. Radcliffe, " Non-linear Genetic Representations, " in Parallel Problem Solving from Nature 2, Elsevier Science Publishers, pp. 259-267, 1992.

[28] A. Wright, " Genetic algorithms for real parameter optimization, " in Foundations of Genetic Algorithms, G. Rawlins Ed., Morgan Kaufman, San Mateo, CA, pp. 205-218, 1991.

[29] W. Gropp, E. Lusk and A. Skjellum, "Using MPI: Portable Parallel Programming with Message-Passing Interface, " MIT Press, Cambridge, 1994.

[30] M.G. Moharam, E. B. Grann, D. A. Pommet, and T. K. Gaylord, " Formulation for stable and efficient representation of the rigorous coupled-wave analysis of binary gratings, " J. Opt. Soc. Am. A vol. 12, no. 5, pp. 1068-1076, May 1995.

[31] M. G. Moharam, D. A. Pommet, E. B. Grann, and T. K. Gaylord, "Stable implementation of the rigorous coupled-wave analysis for surface-relief gratings: enhanced transmittance matrix approach," J. Opt. Soc. Am. A vol. 12, no. 5, pp. 1077-1086, May 1995.

[32] N. Amitay, V. Galindo and C. Wu, " Theory and analysis of phased array antennas, " New York, Wiley, pp. 310-313, 1972.

[33] S. Chakrabarti, K.R. Demarest, and E.K. Miller, "An extended frequency-domain Prony's method for transfer function parameter estimation, " *Intl. J. Numerical Modeling:Electronic Networks, Devices and Fields,* vol. 6, pp.269-281, 1993.

[34] C. Zuffada, T. Cwik and C. Ditchman, " Synthesis of novel all-dielectric grating filters using genetic algorithms, " submitted for publication in *IEEE Trans. Antennas and Propagation,* 1997.

[35] R. Jorgenson and R. Mittra, " Efficient calculation of the free-space periodic Green's function, " *IEEE Trans. Antennas and Propagation,* vol. 38, pp. 633-642, May 1990.

```
include "pgapackf.h"
external evaluate
integer ctx
ctx = PGACreate (PGA_DATATYPE_BINARY, 100, PGA_MAXIMIZE)
call  PGASetUp  (ctx                    )
call  PGARun    (ctx, evaluate          )
call  PGADestroy(ctx                    )
stop
end

double precision function evaluate (ctx, p, pop)
include "pgapackf.h"
integer ctx, p, pop, i, bit, nbits, stringlen
stringlen = PGAGetStringLength(ctx)
nbits    = 0
do i=1, stringlen
  bit = PGAGetBinaryAllele(ctx, p, pop, i)
  if (bit .eq. 1) then
    nbits = nbits + 1
  endif
enddo
evaluate = dble(nbits)
return
end
```

Figure 1. Fortran Program for the ‘Maxbit’ Example. The user provides the function *evaluate* to specify the merit of a candidate solution.

```
#include <pgapack.h>
#define  NUM_LAYERS     4     /* composite material has 4 layers    */
#define  NUM_MATERIALS  6     /* choose among 6 materials per layer */

typedef struct {
   int    layer[NUM_LAYERS];  /* material type      */
   double thick[NUM_LAYERS];  /* material thickness */
} composite;

double     weight        (double *, int *);
double     Evaluate      (PGAContext *, int, int);
int        CompMutation    (PGAContext *, int, int, double);

int main(int argc, char **argv) {
   PGAContext *ctx;
   int maxiter;
   ctx  =  PGACreate(&argc,  argv,  PGA_DATATYPE_USER,  2*NUM_LAYERS,
PGA_MINIMIZE);
   PGASetUserFunction  (ctx, PGA_USERFUNCTION_MUTATION, CompMutation);
   PGASetUp          (ctx);
   PGARun           (ctx, Evaluate);
   PGADestroy        (ctx);
   return;
}
```

Figure 2. Main Program for ‘Composite’ Data Type, a user-defined data type which can be integrated with PGAPACK.

```
int CompMutation(PGAContext *ctx, int p, int pop, double mr) {
  composite *comp;
  int i, count = 0;

  comp = (composite *)PGAGetIndividual(ctx, p, pop)->chrom;
  for (i = 0; i < NUM_LAYERS; i++)
    if (PGARandomFlip(ctx, mr)) {
      if (PGARandomFlip(ctx, 0.5))
        comp->thickness[i] += PGARandomGaussian(ctx,0.,1.);
      count++;
    }
  for (i = 0; i < NUM_LAYERS; i++)
    if (PGARandomFlip(ctx, mr)) {
      comp->layer[i] = PGARandomUniform(1,NUM_MATERIALS);
    }
  return (count);
}
```

Figure 3. One possible definition of the mutation operation for ‘Composite’ Data Type.

```
double Evaluate(PGAContext *ctx, int p, int pop) {
  int i, j;
  int   l[NUM_LAYERS];
  double t[NUM_LAYERS];
```

```
    composite *comp;


    comp = (composite *)PGAGetIndividual(ctx, p, pop)->chrom;
    for (i = 0; i < 6; i++)
       l[i] = comp->layer[i];
    for (i = 0; i < 6; i++)
       t[i] = comp->thickness[i];
    return ( weight(l,t) );
}
```

Figure 4. Example evaluation Function for Composite Data Type.

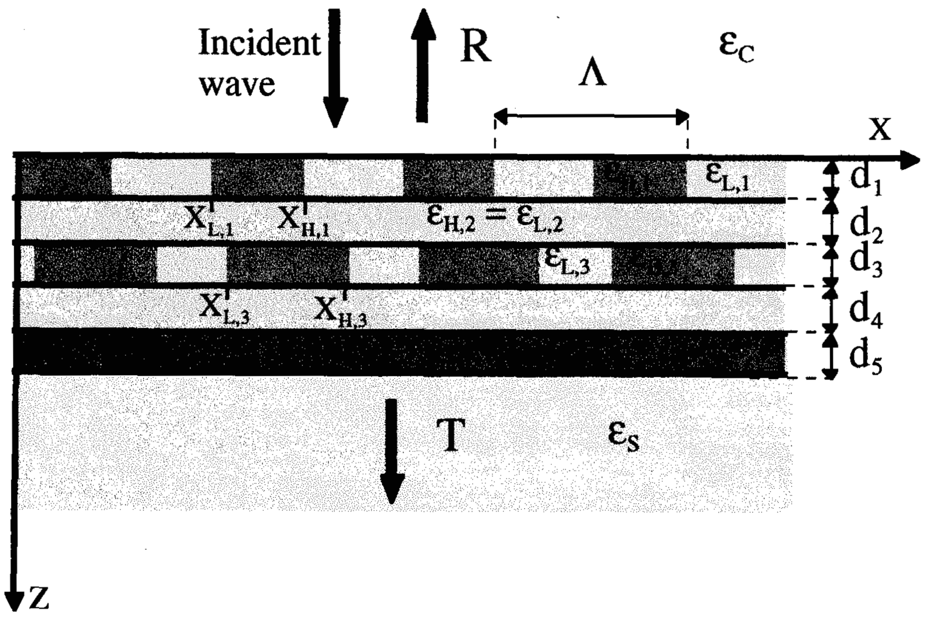


Fig. 5 Multilayer stack of grating and homogeneous layers used as narrowband filter.

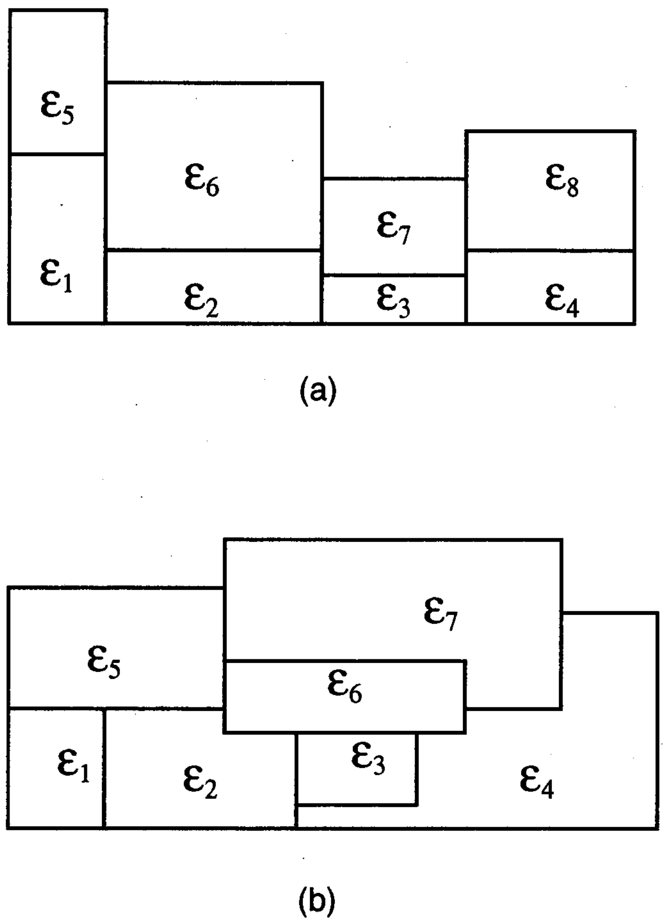


Figure 6. Periodic cell corresponding to two possible filter realizations, alternative to that of Fig. 5, which can be synthesized with the scattering model of III.2.

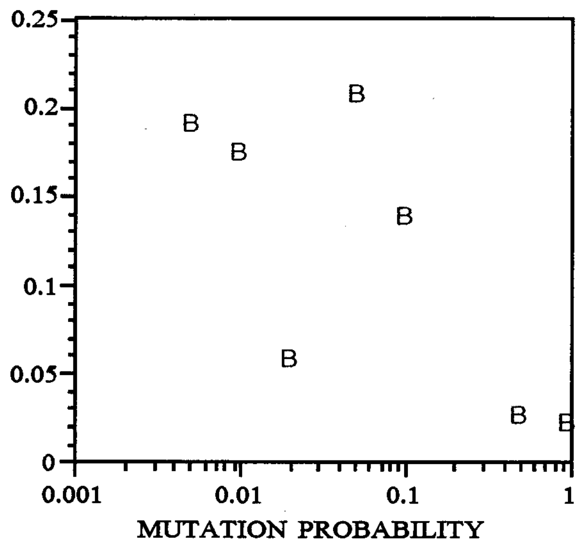
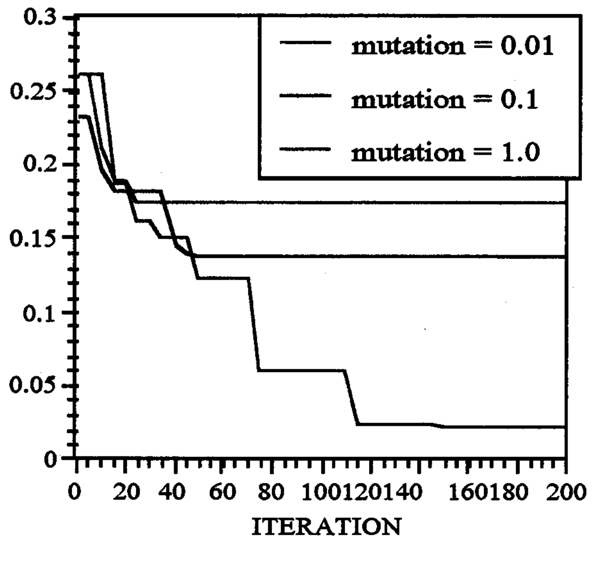
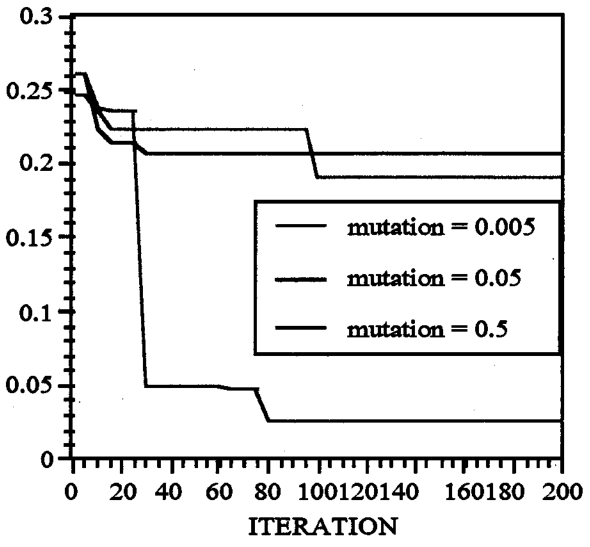


Figure 7. Residual as a function of mutation probability for real encoding. PGAPACK parameters; population = 500, generations = 200, replacement per generation = 200, gaussian mutation with $\sigma = 0.1$, uniform crossover with prob. = 0.8.



(8a)



(8b)

Figure 8. Convergence history for real encoding with different mutation probability. PGAPACK parameters; population = 500, replacement per generation 200, uniform crossover with prob. = 0.8.

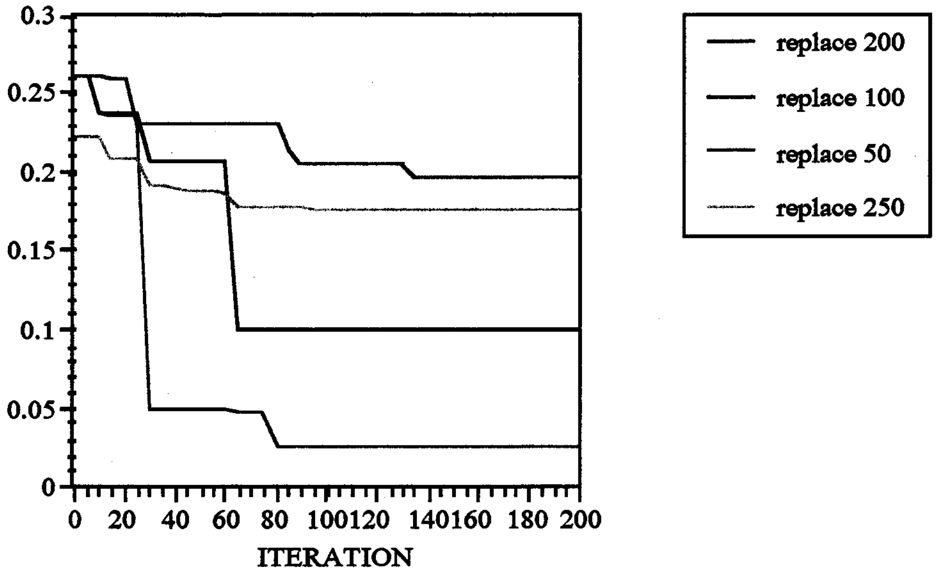


Figure 9. Convergence history for real encoding with different replacement values. PGAPACK parameters; population = 500, gaussian mutation with probability 0.5 and $\sigma = 0.1$, uniform crossover with prob. = 0.8.

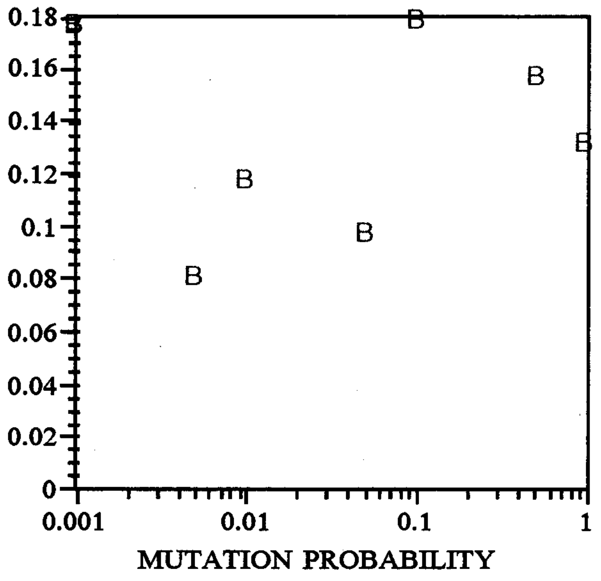
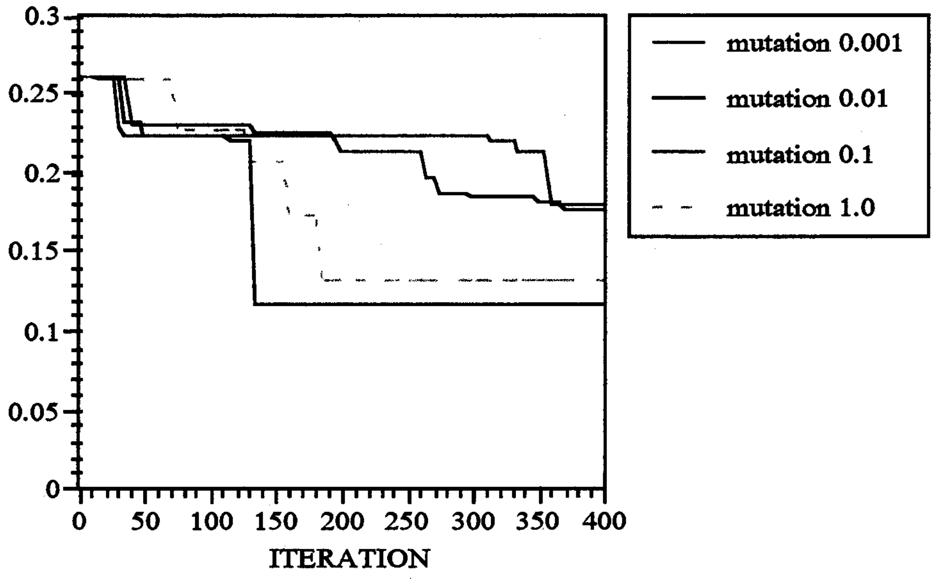
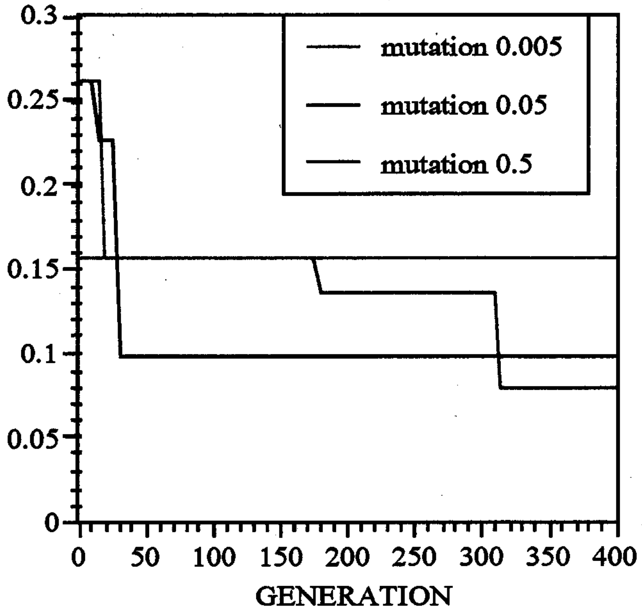


Figure 10. Residual as a function of mutation probability for binary encoding. PGAPACK parameters; population = 500, generations = 400, replacement per generation = 50, uniform crossover with prob. = 0.8.



(11a)



(11b)

Figure 11. Convergence history for binary encoding with different mutation probability. PGAPACK parameters; population = 500, replacement per generation 50, uniform crossover with prob. = 0.8.

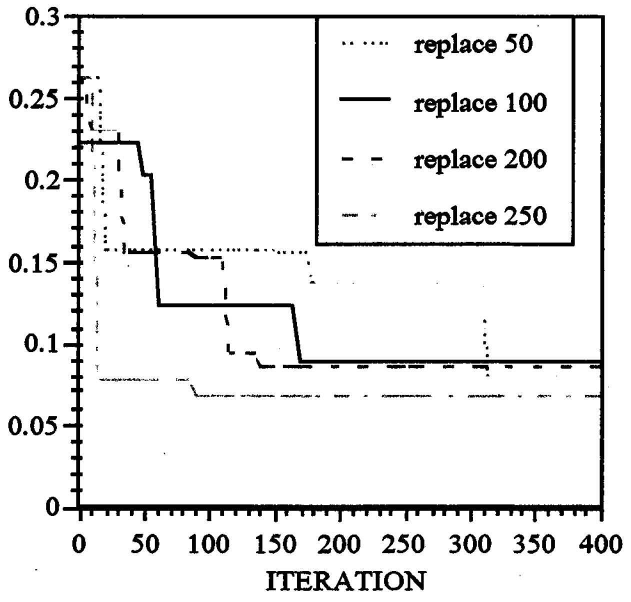


Figure 12. Convergence history for binary encoding with different replacement values. PGAPACK parameters; population = 500, mutation with probability 0.005, uniform crossover with prob. = 0.8.

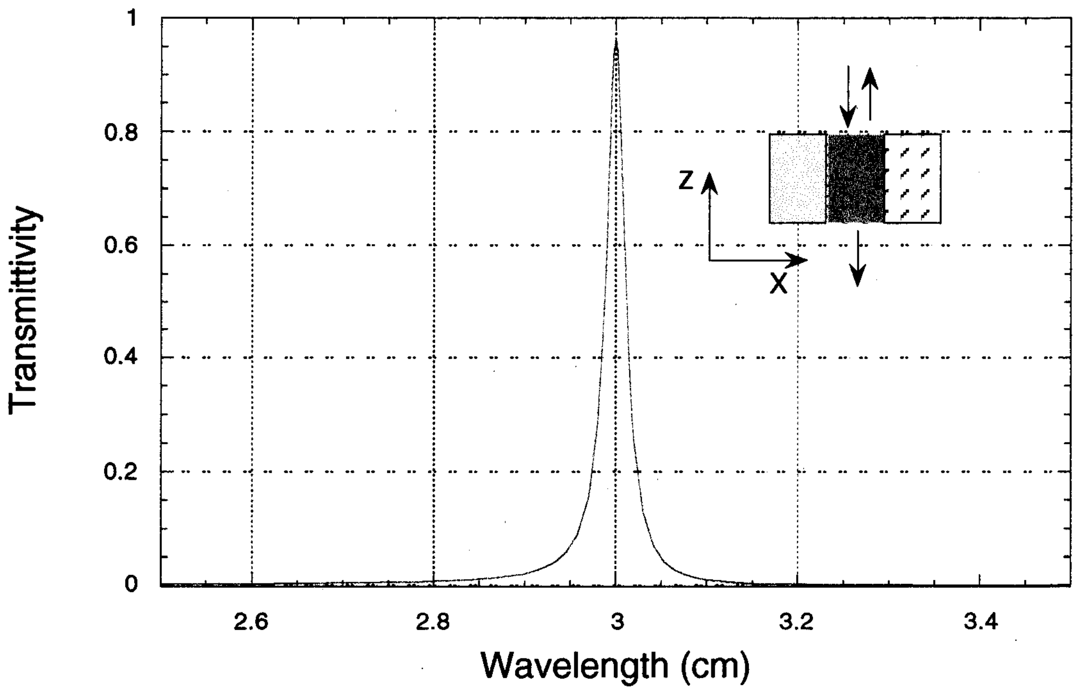


Figure 13. Transmittivity response for three-material single-grating filter designed to satisfy a Lorentzian lineshape. Cell size is Tx = 2.2 cm, thickness = 0.9 cm. Boundary locations are at $x_{12} = 1.15$, $x_{23} = 1.78$ cm. Dielectric constants of materials from left to right are $\varepsilon_1 = 2.498$, $\varepsilon_2 = 7.939$, $\varepsilon_3 = 10$.

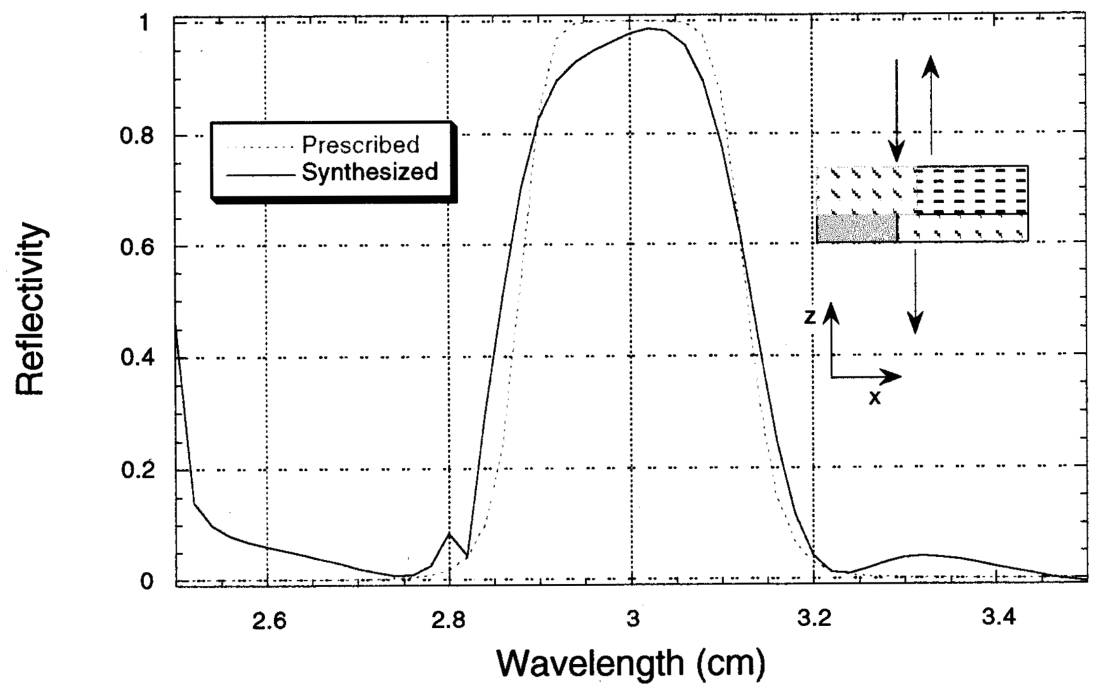


Figure 4. Reflectivity response for double-grating filter designed to satisfy a Butterworth lineshape of the fourth order. Cell size is Tx = 2.5 cm, thickness = 2.7 cm. Boundary locations along x are at $x_{12} = 1.5$ cm (bottom grating) and $x_{34} = 1.25$ cm (top grating). Thickness of bottom grating is 0.96 cm. Dielectric constants of the four materials are $\varepsilon_1 = 1.319$, $\varepsilon_2 = 9.172$, $\varepsilon_3 = 3.638$, $\varepsilon_4 = 4.113$.